

Why Passwords Keep Losing: MFA, Passkeys, and Account Takeover

Wed, Aug 5, 9:00 AM EDT · <https://scottslab.io/posts/why-passwords-keep-losing-mfa-passkeys>

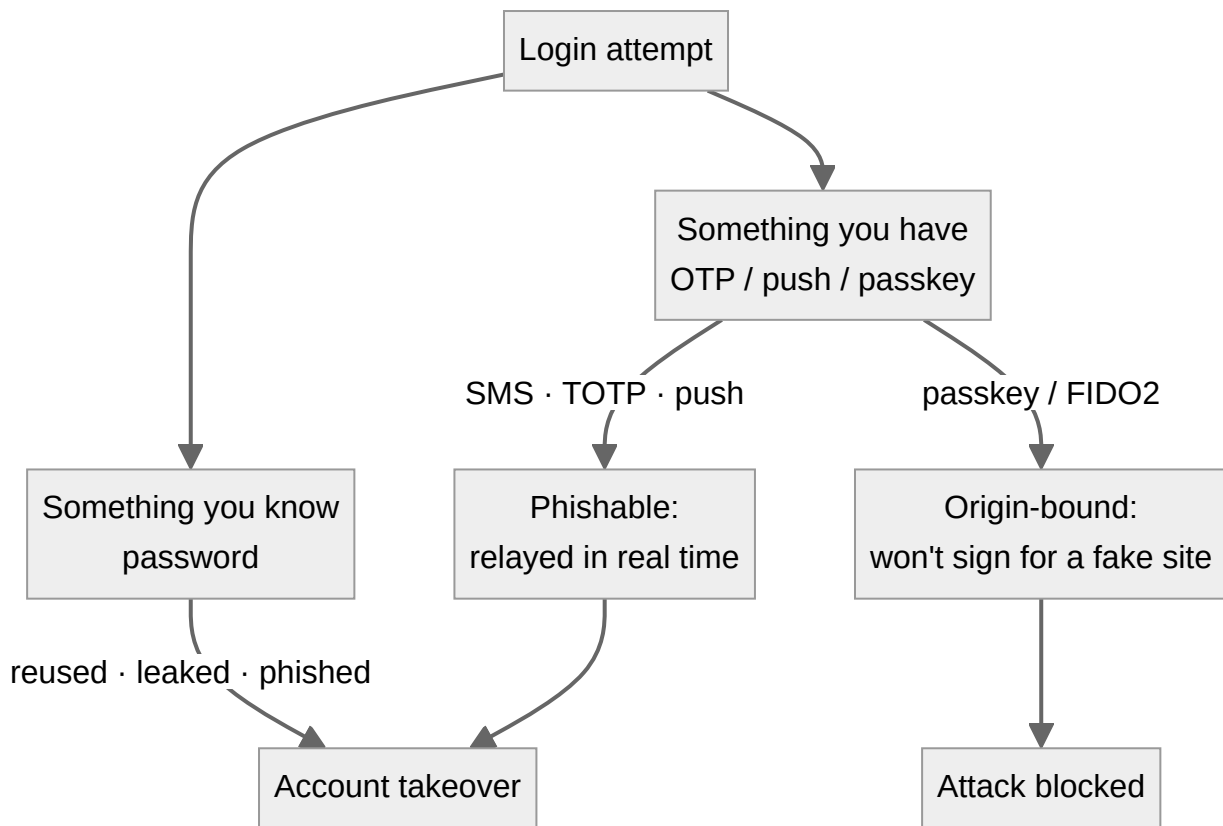


TL;DR

Most account takeovers don't crack anything; they reuse a leaked password or phish it on a convincing fake page. MFA helps, but not all factors are equal. SMS and TOTP are phishable, push invites fatigue attacks, and only passkeys/FIDO2 fix phishing structurally by binding the credential to the site's origin. The next place worth spending attention is making account recovery as phishing-resistant as the login itself.

Most of the account takeovers I've worked never involved cracking anything. Nobody brute-forced a strong password. Someone reused a password that leaked in an unrelated breach, or typed it into a convincing login page, and that was the whole attack. The password worked exactly as designed. It proved knowledge of a secret. The secret just hadn't been secret for months.

That's the thing worth internalizing before you argue about password length policies: a password is a shared secret, and shared secrets leak. Rotation, complexity rules, the little strength meter: they're all patching a model that breaks the moment the secret is copied. So the useful question isn't "how do I make the password stronger," it's "how few things break when the password is already in the attacker's hands."

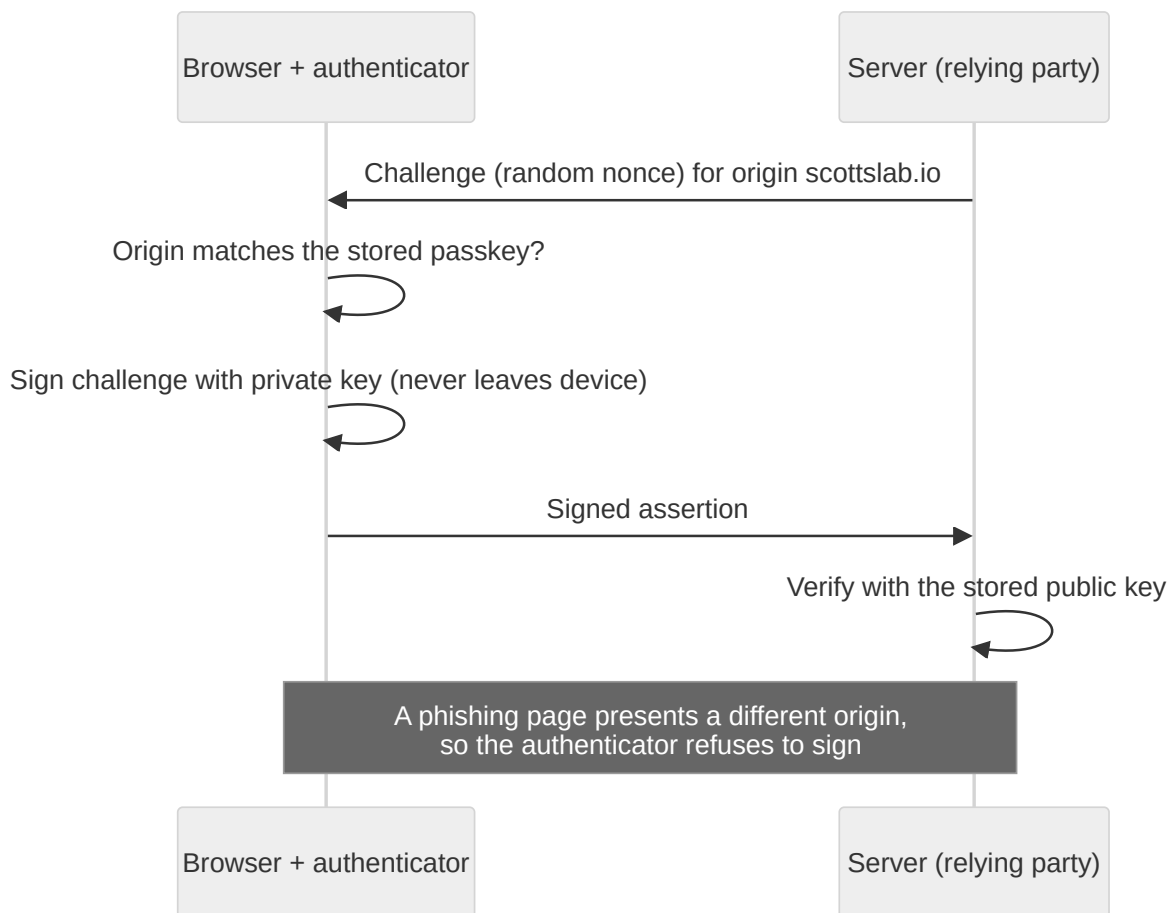


Why Passwords Keep Losing: MFA, Passkeys, and Account Takeover

MFA is the first real answer, because it stops you standing on a single leaked secret. The catch is that not all second factors are equal, and the industry spent a decade pretending they were. SMS codes and TOTP apps both help against pure credential reuse, but both are phishable. A fake login page just asks for the code too and relays it in real time. I've watched exactly that in incident timelines: password and OTP harvested on the same fake page, replayed within seconds. The second factor was real; it just wasn't resistant to the attack that mattered.

Push approvals have their own failure mode: the plain "approve?" prompt trained people to tap yes, and attackers learned to spam prompts at 3am until someone did. Number-matching helped, but any factor that leans on a tired human making a judgment call is a control with a snooze button.

Passkeys are the first factor that fixes the phishing problem structurally instead of asking the user to be careful. WebAuthn/FIDO2 binds the credential to the site's origin and never sends a reusable secret. The browser signs a challenge with a private key that never leaves the device, and it simply won't sign for the wrong origin. That last part is the whole game: a pixel-perfect phishing page can't get a valid assertion, because the domain doesn't match. It isn't "a better password." It's a different model: proof of possession instead of proof of knowledge.



Why Passwords Keep Losing: MFA, Passkeys, and Account Takeover

None of this makes passwords vanish tomorrow. Recovery flows, legacy systems, and the account-bootstrap problem keep them around, and a passkey is only as strong as the recovery path behind it. An attacker who can reset his way back to an SMS code has routed around the good control entirely. That's where I'd spend attention now: not on password complexity, but on making the fallbacks as phishing-resistant as the happy path. The front door is finally getting good locks. The attacks are already moving to the windows.